Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention

Zhang, Biao ; Titov, Ivan ; Sennrich, Rico

Abstract: The general trend in NLP is towards increasing model capacity and performance via deeper neural networks. However, simply stacking more layers of the popular Transformer architecture for machine translation results in poor convergence and high computational overhead. Our empirical analysis suggests that convergence is poor due to gradient vanishing caused by the interaction between residual connection and layer normalization. We propose depth-scaled initialization (DS-Init), which decreases parameter variance at the initialization stage, and reduces output variance of residual connections so as to ease gradient back-propagation through normalization layers. To address computational cost, we propose a merged attention sublayer (MAtt) which combines a simplified average-based self-attention sublayer and the encoder-decoder attention sublayer on the decoder side. Results on WMT and IWSLT translation tasks with five translation directions show that deep Transformers with DS-Init and MAtt can substantially outperform their base counterpart in terms of BLEU (+1.1 BLEU on average for 12-layer models), while matching the decoding speed of the baseline model thanks to the efficiency improvements of MAtt.

Originally published at:
Zhang, Biao; Titov, Ivan; Sennrich, Rico (2019). Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP), Hong Kong, 3 November 2019 - 7 November 2019. Association for Computational Linguistics, 897-908.

Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention

Biao Zhang[1] **Ivan Titov**[1,2] **Rico Sennrich**[3,1]

[1]School of Informatics, University of Edinburgh
[2]ILLC, University of Amsterdam
[3]Institute of Computational Linguistics, University of Zurich
B.Zhang@ed.ac.uk, ititov@inf.ed.ac.uk, sennrich@cl.uzh.ch

Abstract

The general trend in NLP is towards increasing model capacity and performance via deeper neural networks. However, simply stacking more layers of the popular Transformer architecture for machine translation results in poor convergence and high computational overhead. Our empirical analysis suggests that convergence is poor due to *gradient vanishing* caused by the interaction between residual connections and layer normalization. We propose depth-scaled initialization (DS-Init), which decreases parameter variance at the initialization stage, and reduces output variance of residual connections so as to ease gradient back-propagation through normalization layers. To address computational cost, we propose a merged attention sublayer (MAtt) which combines a simplified average-based self-attention sublayer and the encoder-decoder attention sublayer on the decoder side. Results on WMT and IWSLT translation tasks with five translation directions show that deep Transformers with DS-Init and MAtt can substantially outperform their base counterpart in terms of BLEU (+1.1 BLEU on average for 12-layer models), while matching the decoding speed of the baseline model thanks to the efficiency improvements of MAtt.[1]

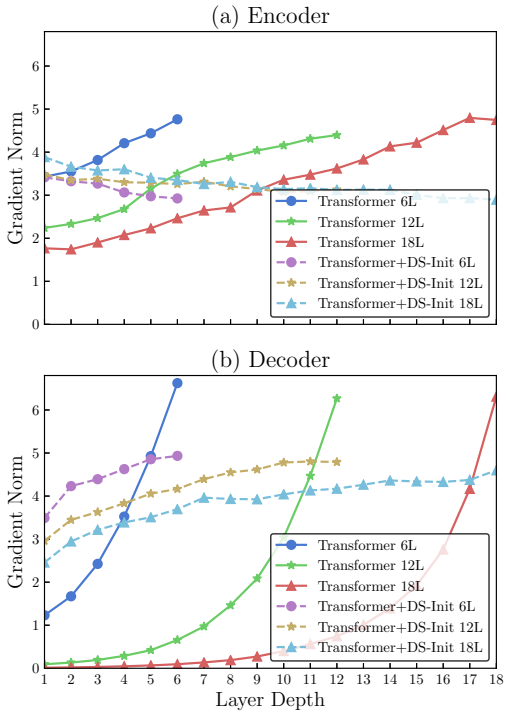


Figure 1: Gradient norm (y-axis) of each encoder layer (top) and decoder layer (bottom) in Transformer with respect to layer depth (x-axis). Gradients are estimated with ∼3k target tokens at the beginning of training. "*DS-Init*": the proposed depth-scaled initialization. "*6L*": 6 layers. Solid lines indicate the vanilla Transformer, and dashed lines denote our proposed method. During back-propagation, gradients in Transformer gradually vanish from high layers to low layers.

1 Introduction

The capability of deep neural models of handling complex dependencies has benefited various artificial intelligence tasks, such as image recognition where test error was reduced by scaling VGG nets (Simonyan and Zisserman, 2015) up to hundreds of convolutional layers (He et al., 2015). In NLP, deep self-attention networks have enabled large-scale pretrained language models such as BERT (Devlin et al., 2019) and GPT (Radford et al., 2018) to boost state-of-the-art (SOTA) performance on downstream applications. By contrast, though neural machine translation (NMT) gained encouraging improvement when shifting from a shallow architecture (Bahdanau et al., 2015) to deeper ones (Zhou et al., 2016; Wu et al., 2016; Zhang et al., 2018; Chen et al., 2018), the Transformer (Vaswani et al., 2017), a currently SOTA architecture, achieves best results with merely 6 encoder and decoder layers, and no gains were reported by Vaswani et al. (2017) from

[1]Source code for reproduction is available at https://github.com/bzhangGo/zero

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 897–908, Hong Kong, China, November 3–7, 2019.

further increasing its depth on standard datasets.

We start by analysing why the Transformer does not scale well to larger model depth. We find that the architecture suffers from gradient vanishing as shown in Figure 1, leading to poor convergence. An in-depth analysis reveals that the Transformer is not norm-preserving due to the involvement of and the interaction between residual connection (RC) (He et al., 2015) and layer normalization (LN) (Ba et al., 2016).

To address this issue, we propose depth-scaled initialization (DS-Init) to improve norm preservation. We ascribe the gradient vanishing to the large output variance of RC and resort to strategies that could reduce it without model structure adjustment. Concretely, DS-Init scales down the variance of parameters in the l-th layer with a discount factor of $\frac{1}{\sqrt{l}}$ at the initialization stage alone, where l denotes the layer depth starting from 1. The intuition is that parameters with small variance in upper layers would narrow the output variance of corresponding RCs, improving norm preservation as shown by the dashed lines in Figure 1. In this way, DS-Init enables the convergence of deep Transformer models to satisfactory local optima.

Another bottleneck for deep Transformers is the increase in computational cost for both training and decoding. To combat this, we propose a merged attention network (MAtt). MAtt simplifies the decoder by replacing the separate self-attention and encoder-decoder attention sublayers with a new sublayer that combines an efficient variant of average-based self-attention (AAN) (Zhang et al., 2018) and the encoder-decoder attention. We simplify the AAN by reducing the number of linear transformations, reducing both the number of model parameters and computational cost. The merged sublayer benefits from parallel calculation of (average-based) self-attention and encoder-decoder attention, and reduces the depth of each decoder block.

We conduct extensive experiments on WMT and IWSLT translation tasks, covering five translation tasks with varying data conditions and translation directions. Our results show that deep Transformers with DS-Init and MAtt can substantially outperform their base counterpart in terms of BLEU (+1.1 BLEU on average for 12-layer models), while matching the decoding speed of the baseline model thanks to the efficiency improvements of MAtt.

Our contributions are summarized as follows:

- We analyze the vanishing gradient issue in the Transformer, and identify the interaction of residual connections and layer normalization as its source.
- To address this problem, we introduce depth-scaled initialization (DS-Init).
- To reduce the computational cost of training deep Transformers, we introduce a merged attention model (MAtt). MAtt combines a simplified average-attention model and the encoder-decoder attention into a single sublayer, allowing for parallel computation.
- We conduct extensive experiments and verify that deep Transformers with DS-Init and MAtt improve translation quality while preserving decoding efficiency.

2 Related Work

Our work aims at improving translation quality by increasing model depth. Compared with the single-layer NMT system (Bahdanau et al., 2015), deep NMT models are typically more capable of handling complex language variations and translation relationships via stacking multiple encoder and decoder layers (Zhou et al., 2016; Wu et al., 2016; Britz et al., 2017; Chen et al., 2018), and/or multiple attention layers (Zhang et al., 2018). One common problem for the training of deep neural models are vanishing or exploding gradients. Existing methods mainly focus on developing novel network architectures so as to stabilize gradient back-propagation, such as the fast-forward connection (Zhou et al., 2016), the linear associative unit (Wang et al., 2017), or gated recurrent network variants (Hochreiter and Schmidhuber, 1997; Gers and Schmidhuber, 2001; Cho et al., 2014; Di Gangi and Federico, 2018). In contrast to the above recurrent network based NMT models, recent work focuses on feed-forward alternatives with more smooth gradient flow, such as convolutional networks (Gehring et al., 2017) and self-attention networks (Vaswani et al., 2017).

The Transformer represents the current SOTA in NMT. It heavily relies on the combination of residual connections (He et al., 2015) and layer normalization (Ba et al., 2016) for convergence. Nevertheless, simply extending this model with more layers results in gradient vanishing due to the interaction of RC and LN (see Section 4). Recent work has proposed methods to train deeper

Transformer models, including a rescheduling of RC and LN (Vaswani et al., 2018), the transparent attention model (Bapna et al., 2018) and the stochastic residual connection (Pham et al., 2019). In contrast to these work, we identify the large output variance of RC as the source of gradient vanishing, and employ scaled initialization to mitigate it without any structure adjustment. The effect of careful initialization on boosting convergence was also investigated and verified in previous work (Zhang et al., 2019; Child et al., 2019; Devlin et al., 2019; Radford et al., 2018).

The merged attention network falls into the category of simplifying the Transformer so as to shorten training and/or decoding time. Methods to improve the Transformer's running efficiency range from algorithmic improvements (Junczys-Dowmunt et al., 2018), non-autoregressive translation (Gu et al., 2018; Ghazvininejad et al., 2019) to decoding dependency reduction such as average attention network (Zhang et al., 2018) and blockwise parallel decoding (Stern et al., 2018). Our MAtt builds upon the AAN model, further simplifying the model by reducing the number of linear transformations, and combining it with the encoder-decoder attention. In work concurrent to ours, So et al. (2019) propose the evolved Transformer which, based on automatic architecture search, also discovered a parallel structure of self-attention and encoder-decoder attention.

3 Background: Transformer

Given a source sequence $\mathbf{X} = \{x_1, x_2, \ldots, x_n\} \in \mathbb{R}^{n \times d}$, the Transformer predicts a target sequence $\mathbf{Y} = \{y_1, y_2, \ldots, y_m\}$ under the encoder-decoder framework. Both the encoder and the decoder in the Transformer are composed of attention networks, functioning as follows:

$$\begin{aligned} \textsc{Att}(\mathbf{Z}_x, \mathbf{Z}_y) &= \left[\text{softmax}(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d}})\mathbf{V}\right]\mathbf{W}_o \\ \mathbf{Q}, \mathbf{K}, \mathbf{V} &= \mathbf{Z}_x\mathbf{W}_q, \mathbf{Z}_y\mathbf{W}_k, \mathbf{Z}_y\mathbf{W}_v, \end{aligned} \quad (1)$$

where $\mathbf{Z}_x \in \mathbb{R}^{I \times d}$ and $\mathbf{Z}_y \in \mathbb{R}^{J \times d}$ are input sequence representations of length I and J respectively, $\mathbf{W}_* \in \mathbb{R}^{d \times d}$ denote weight parameters. The attention network can be further enhanced with multi-head attention (Vaswani et al., 2017).

Formally, the encoder stacks L identical layers, each including a self-attention sublayer (Eq. 2) and a point-wise feed-forward sublayer (Eq. 3):

$$\bar{\mathbf{H}}^l = \textsc{Ln}\left(\textsc{Rc}\left(\mathbf{H}^{l-1}, \textsc{Att}(\mathbf{H}^{l-1}, \mathbf{H}^{l-1})\right)\right) \quad (2)$$

$$\mathbf{H}^l = \textsc{Ln}\left(\textsc{Rc}\left(\bar{\mathbf{H}}^l, \textsc{Ffn}(\bar{\mathbf{H}}^l)\right)\right). \quad (3)$$

$\mathbf{H}^l \in \mathbb{R}^{n \times d}$ denotes the sequence representation of the l-th encoder layer. Input to the first layer $\mathbf{H}^0$ is the element-wise addition of the source word embedding $\mathbf{X}$ and the corresponding positional encoding. $\textsc{Ffn}(\cdot)$ is a two-layer feed-forward network with a large intermediate representation and ReLU activation function. Each encoder sublayer is wrapped with a residual connection (Eq. 4), followed by layer normalization (Eq. 5):

$$\textsc{Rc}(\mathbf{z}, \mathbf{z}') = \mathbf{z} + \mathbf{z}', \quad (4)$$

$$\textsc{Ln}(\mathbf{z}) = \frac{\mathbf{z} - \mu}{\sigma} \odot \mathbf{g} + \mathbf{b}, \quad (5)$$

where $\mathbf{z}$ and $\mathbf{z}'$ are input vectors, and $\odot$ indicates element-wise multiplication. μ and σ denote the mean and standard deviation statistics of vector $\mathbf{z}$. The normalized $\mathbf{z}$ is then re-scaled and re-centered by trainable parameters $\mathbf{g}$ and $\mathbf{b}$ individually.

The decoder also consists of L identical layers, each of them extends the encoder sublayers with an encoder-decoder attention sublayer (Eq. 7) to capture translation alignment from target words to relevant source words:

$$\tilde{\mathbf{S}}^l = \textsc{Ln}\left(\textsc{Rc}\left(\mathbf{S}^{l-1}, \textsc{Att}(\mathbf{S}^{l-1}, \mathbf{S}^{l-1})\right)\right) \quad (6)$$

$$\bar{\mathbf{S}}^l = \textsc{Ln}\left(\textsc{Rc}\left(\tilde{\mathbf{S}}^l, \textsc{Att}(\tilde{\mathbf{S}}^l, \mathbf{H}^L)\right)\right) \quad (7)$$

$$\mathbf{S}^l = \textsc{Ln}\left(\textsc{Rc}\left(\bar{\mathbf{S}}^l, \textsc{Ffn}(\bar{\mathbf{S}}^l)\right)\right). \quad (8)$$

$\mathbf{S}^l \in \mathbb{R}^{m \times d}$ is the sequence representation of the l-th decoder layer. Input $\mathbf{S}^0$ is defined similar to $\mathbf{H}^0$. To ensure auto-regressive decoding, the attention weights in Eq. 6 are masked to prevent attention to future target tokens.

The Transformer's parameters are typically initialized by sampling from a uniform distribution:

$$\mathbf{W} \in \mathbb{R}^{d_i \times d_o} \sim \mathcal{U}(-\gamma, \gamma), \gamma = \sqrt{\frac{6}{d_i + d_o}}, \quad (9)$$

where d_i and d_o indicate input and output dimension separately. This initialization has the advantage of maintaining activation variances and back-propagated gradients variance and can help train deep neural networks (Glorot and Bengio, 2010).

4 Vanishing Gradient Analysis

One natural way to deepen Transformer is simply enlarging the layer number L. Unfortunately, Figure 1 shows that this would give rise to gradient vanishing on both the encoder and the decoder at the lower layers, and that the case on the decoder side is worse. We identified a structural problem in the Transformer architecture that gives rise to this issue, namely the interaction of RC and LN, which we will here discuss in more detail.

Given an input vector $\mathbf{z} \in \mathbb{R}^d$, let us consider the general structure of RC followed by LN:

$$\mathbf{r} = \textsc{Rc}\left(\mathbf{z}, f(\mathbf{z})\right), \tag{10}$$

$$\mathbf{o} = \textsc{Ln}\left(\mathbf{r}\right), \tag{11}$$

where $\mathbf{r}, \mathbf{o} \in \mathbb{R}^d$ are intermediate outputs. $f(\cdot)$ represents any neural network, such as recurrent, convolutional or attention network, etc. Suppose during back-propagation, the error signal at the output of LN is δ_o. Contributions of RC and LN to the error signal are as follows:

$$\delta_r = \frac{\partial \mathbf{o}}{\partial \mathbf{r}}\delta_o = \text{diag}(\frac{\mathbf{g}}{\sigma_r})(\mathbf{I} - \frac{1 - \bar{\mathbf{r}}\bar{\mathbf{r}}^T}{d})\delta_o \tag{12}$$

$$\delta_z = \frac{\partial \mathbf{r}}{\partial \mathbf{z}}\delta_r = (1 + \frac{\partial f}{\partial \mathbf{z}})\delta_r, \tag{13}$$

where $\bar{\mathbf{r}}$ denotes the normalized input. $\mathbf{I}$ is the identity matrix and $\text{diag}(\cdot)$ establishes a diagonal matrix from its input. The resulting δ_r and δ_z are error signals arrived at output $\mathbf{r}$ and $\mathbf{z}$ respectively.

We define the change of error signal as follows:

$$\beta = \beta_{\textsc{Ln}} \cdot \beta_{\textsc{Rc}} = \frac{\|\delta_z\|_2}{\|\delta_r\|_2} \cdot \frac{\|\delta_r\|_2}{\|\delta_o\|_2}, \tag{14}$$

where β (or model ratio), $\beta_{\textsc{Ln}}$ (or LN ratio) and $\beta_{\textsc{Rc}}$ (or RC ratio) measure the gradient norm ratio[2] of the whole residual block, the layer normalization and the residual connection respectively. Informally, a neural model should preserve the gradient norm between layers ($\beta \approx 1$) so as to allow training of very deep models (see Zaeemzadeh et al., 2018).

We resort to empirical evidence to analyze these ratios. Results in Table 1 show that LN weakens error signal ($\beta_{\textsc{Ln}} < 1$) but RC strengthens it ($\beta_{\textsc{Rc}} > 1$). One explanation about LN's decay effect is the large output variance of RC ($\text{Var}(\mathbf{r}) >$ 1) which negatively affects δ_r as shown in Eq. 12. By contrast, the short-cut in RC ensures that the error signal at higher layer δ_r can always be safely carried on to lower layer no matter how complex $\frac{\partial f}{\partial \mathbf{z}}$ would be as in Eq. 13, increasing the ratio.

[2]Model gradients depend on both error signal and layer activation. Reduced/enhanced error signal does not necessarily result in gradient vanishing/explosion, but strongly contributes to it.

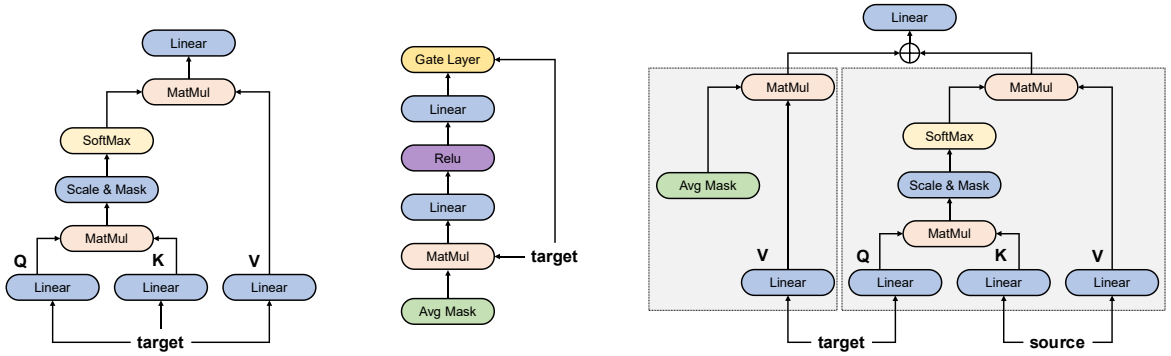
Method	Module		Self	Cross	FFN
Base	Enc	$\beta_{\textsc{Ln}}$	0.86	-	0.84
		$\beta_{\textsc{Rc}}$	1.22	-	1.10
		β	1.05	-	0.93
		$\text{Var}(\mathbf{r})$	1.38	-	1.40
	Dec	$\beta_{\textsc{Ln}}$	0.82	0.74	0.84
		$\beta_{\textsc{Rc}}$	1.21	1.00	1.11
		β	0.98	0.74	0.93
		$\text{Var}(\mathbf{r})$	1.48	1.84	1.39
Ours	Enc	$\beta_{\textsc{Ln}}$	0.96	-	0.95
		$\beta_{\textsc{Rc}}$	1.04	-	1.02
		β	1.02	-	0.98
		$\text{Var}(\mathbf{r})$	1.10	-	1.10
	Dec	$\beta_{\textsc{Ln}}$	0.95	0.94	0.94
		$\beta_{\textsc{Rc}}$	1.05	1.00	1.02
		β	1.10	0.95	0.98
		$\text{Var}(\mathbf{r})$	1.13	1.15	1.11

Table 1: Empirical measure of output variance $\text{Var}(\mathbf{r})$ of RC and error signal change ratio $\beta_{\textsc{Ln}}$, $\beta_{\textsc{Rc}}$ and β (Eq. 14) averaged over 12 layers. These values are estimated with $\sim$3k target tokens at the beginning of training using 12-layer Transformer. "*Base*": the baseline Transformer. "*Ours*": the Transformer with DS-Init. *Enc* and *Dec* stand for encoder and decoder respectively. *Self*, *Cross* and *FFN* indicate the self-attention, encoder-decoder attention and the feed-forward sublayer respectively.

5 Depth-Scaled Initialization

Results on the model ratio show that self-attention sublayer has a (near) increasing effect ($\beta > 1$) that intensifies error signal, while feed-forward sublayer manifests a decreasing effect ($\beta < 1$). In particular, though the encoder-decoder attention sublayer and the self-attention sublayer share the same attention formulation, the model ratio of the former is smaller. As shown in Eq. 7 and 1, part of the reason is that encoder-decoder attention can only back-propagate gradients to lower layers through the query representation $\mathbf{Q}$, bypassing gradients at the key $\mathbf{K}$ and the value $\mathbf{V}$ to the encoder side. This negative effect explains why the decoder suffers from more severe gradient vanishing than the encoder in Figure 1.

The gradient norm is preserved better through the self-attention layer than the encoder-decoder attention, which offers insights on the successful training of the deep Transformer in BERT (Devlin et al., 2019) and GPT (Radford et al., 2018), where encoder-decoder attention is not involved. However, results in Table 1 also suggests that the self-attention sublayer in the encoder is not strong



(a) Self-Attention (b) AAN (c) Merged attention with simplified AAN

Figure 2: An overview of self-attention, AAN and the proposed merged attention with simplified AAN.

enough to counteract the gradient loss in the feed-forward sublayer. That is why BERT and GPT adopt a much smaller standard deviation (0.02) for initialization, in a similar spirit to our solution.

We attribute the gradient vanishing issue to the large output variance of RC (Eq. 12). Considering that activation variance is positively correlated with parameter variance (Glorot and Bengio, 2010), we propose DS-Init and change the original initialization method in Eq. 9 as follows:

$$\mathbf{W} \in \mathbb{R}^{d_i \times d_o} \sim \mathcal{U}\left(-\gamma\frac{\alpha}{\sqrt{l}}, \gamma\frac{\alpha}{\sqrt{l}}\right), \qquad (15)$$

where α is a hyperparameter in the range of $[0, 1]$ and l denotes layer depth. Hyperparameter α improves the flexibility of our method. Compared with existing approaches (Vaswani et al., 2018; Bapna et al., 2018), our solution does not require modifications in the model architecture and hence is easy to implement.

According to the property of uniform distribution, the variance of model parameters decreases from $\frac{\gamma^2}{3}$ to $\frac{\gamma^2\alpha^2}{3l}$ after applying DS-Init. By doing so, a higher layer would have smaller output variance of RC so that more gradients can flow back. Results in Table 1 suggest that DS-Init narrows both the variance and different ratios to be $\sim$1, ensuring the stability of gradient back-propagation. Evidence in Figure 1 also shows that DS-Init helps keep the gradient norm and slightly increases it on the encoder side. This is because DS-Init endows lower layers with parameters of larger variance and activations of larger norm. When error signals at different layers are of similar scale, the gradient norm at lower layers would be larger. Nevertheless, this increase does not hurt model training based on our empirical observation.

DS-Init is partially inspired by the *Fixup* initialization (Zhang et al., 2019). Both of them try to reduce the output variance of RC. The difference is that Fixup focuses on overcoming gradient explosion cased by consecutive RCs and seeks to enable training without LN but at the cost of carefully handling parameter initialization of each matrix transformation, including manipulating initialization of different bias and scale terms. Instead, DS-Init aims at solving gradient vanishing in deep Transformer caused by the structure of RC followed by LN. We still employ LN to standardize layer activation and improve model convergence. The inclusion of LN ensures the stability and simplicity of DS-Init.

6 Merged Attention Model

With large model depth, deep Transformer unavoidably introduces high computational overhead. This brings about significantly longer training and decoding time. To remedy this issue, we propose a merged attention model for decoder that integrates a simplified average-based self-attention sublayer into the encoder-decoder attention sublayer. Figure 2 highlights the difference.

The AAN model (Figure 2(b)), as an alternative to the self-attention model (Figure 2(a)), accelerates Transformer decoding by allowing decoding in linear time, avoiding the $\mathcal{O}(n^2)$ complexity of the self-attention mechanism (Zhang et al., 2018). Unfortunately, the gating sublayer and the feed-forward sublayer inside AAN reduce the empirical performance improvement. We propose a simplified AAN by removing all matrix computation except for two linear projections:

$$\mathrm{SAAN}(\mathbf{S}^{l-1}) = \left[\mathbf{M}_a(\mathbf{S}^{l-1}\mathbf{W}_v)\right]\mathbf{W}_o, \qquad (16)$$

Dataset	#Src	#Tgt	#Sent	#BPE
WMT14 En-De	116M	110M	4.5M	32K
WMT14 En-Fr	1045M	1189M	36M	32K
WMT18 En-Fi	73M	54M	3.3M	32K
WMT18 Zh-En	510M	576M	25M	32K
IWSLT14 De-En	3.0M	3.2M	159K	30K

Table 2: Statistics for different training datasets. *#Src* and *#Tgt* denote the number of source and target tokens respectively. *#Sent*: the number of bilingual sentences. *#BPE*: the number of merge operations in BPE. *M*: million, *K*: thousand.

where $\mathbf{M}_a$ denotes the average mask matrix for parallel computation (Zhang et al., 2018). This new model is then combined with the encoder-decoder attention as shown in Figure 2(c):

$$\begin{aligned} \text{MATT}(\mathbf{S}^{l-1}) &= \text{SAAN}(\mathbf{S}^{l-1}) + \text{ATT}(\mathbf{S}^{l-1}, \mathbf{H}^{L}) \\ \bar{\mathbf{S}}^{l} &= \text{LN}\left(\text{RC}\left(\mathbf{S}^{l-1}, \text{MATT}(\mathbf{S}^{l-1})\right)\right). \end{aligned} \quad (17)$$

The mapping $\mathbf{W}_o$ is shared for SAAN and ATT. After combination, MAtt allows for the parallelization of AAN and encoder-decoder attention.

7 Experiments

7.1 Datasets and Evaluation

We take WMT14 English-German translation (En-De) (Bojar et al., 2014) as our benchmark for model analysis, and examine the generalization of our approach on four other tasks: WMT14 English-French (En-Fr), IWSLT14 German-English (De-En) (Cettolo et al., 2014), WMT18 English-Finnish (En-Fi) and WMT18 Chinese-English (Zh-En) (Bojar et al., 2018). Byte pair encoding algorithm (BPE) (Sennrich et al., 2016) is used in preprocessing to handle low frequency words. Statistics of different datasets are listed in Table 2.

Except for IWSLT14 De-En task, we collect subword units independently on the source and target side of training data. We directly use the preprocessed training data from the WMT18 website[3] for En-Fi and Zh-En tasks, and use newstest2017 as our development set, newstest2018 as our test set. Our training data for WMT14 En-De and WMT14 En-Fr is identical to previous setups (Vaswani et al., 2017; Wu et al., 2019). We use newstest2013 as development set for WMT14 En-De and newstest2012+2013 for WMT14 En-Fr. Apart from newstest2014 test set[4], we also evaluate our model on all WMT14-18 test sets for WMT14 En-De translation. The settings for IWSLT14 De-En are as in Ranzato et al. (2016), with 7584 sentence pairs for development, and the concatenated dev sets for IWSLT 2014 as test set (tst2010, tst2011, tst2012, dev2010, dev2012).

We report tokenized case-sensitive BLEU (Papineni et al., 2002) for WMT14 En-De and WMT14 En-Fr, and provide detokenized case-sensitive BLEU for WMT14 En-De, WMT18 En-Fi and Zh-En with *sacreBLEU* (Post, 2018)[5]. We also report chrF score for En-Fi translation which was found correlated better with human evaluation (Bojar et al., 2018). Following previous work (Wu et al., 2019), we evaluate IWSLT14 De-En with tokenized case-insensitive BLEU.

7.2 Model Settings

We experiment with both *base* (layer size 512/2048, 8 heads) and *big* (layer size 1024/4096, 16 heads) settings as in Vaswani et al. (2017). Except for the vanilla Transformer, we also compare with the structure that is currently default in tensor2tensor (T2T), which puts layer normalization before residual blocks (Vaswani et al., 2018). We use an in-house toolkit for all experiments.

Dropout is applied to the residual connection (dp_r) and attention weights (dp_a). We share the target embedding matrix with the softmax projection matrix but not with the source embedding matrix. We train all models using Adam optimizer (0.9/0.98 for base, 0.9/0.998 for big) with adaptive learning rate schedule (warm-up step 4K for base, 16K for big) as in (Vaswani et al., 2017) and label smoothing of 0.1. We set α in DS-Init to 1.0. Sentence pairs containing around 25K$\sim$50K (bs) target tokens are grouped into one batch. We use relatively larger batch size and dropout rate for deeper and bigger models for better convergence. We perform evaluation by averaging last 5 checkpoints. Besides, we apply mixed-precision training to all big models. Unless otherwise stated, we train base and big model with 300K maximum steps, and decode sentences using beam search with a beam size of 4 and length penalty of 0.6. Decoding is implemented with cache to save redundant computations. Other settings for specific translation tasks are explained in the individual subsections.

[3] http://www.statmt.org/wmt18/translation-task.html

[4] We use the filtered test set consisting of 2737 sentence pairs. The difference of translation quality on filtered and full test sets is marginal.

[5] Signature BLEU+c.mixed+#.1+s.exp+tok.13a+v.1.2.20

ID	Model	#Param	Test14	△Dec	△Train
1	Base 6 layers $dp_a = dp_r = 0.1, bs = 25K$	72.3M	27.59 (26.9)	62.26/1.00×	0.105/1.00×
2	1 + T2T	72.3M	27.20 (26.5)	68.04/0.92×	0.105/1.00×
3	1 + DS-Init	72.3M	27.50 (26.8)	⋆/1.00×	⋆/1.00×
4	1 + MAtt	66.0M	27.49 (26.8)	40.51/1.54×	0.094/1.12×
5	1 + MAtt + DS-Init	66.0M	27.35 (26.8)	40.84/1.52×	0.094/1.12×
6	1 + MAtt with self-attention	72.3M	27.41 (26.7)	60.25/1.03×	0.105/1.00×
7	1 + MAtt with original AAN	72.2M	27.36 (26.7)	46.13/1.35×	0.098/1.07×
8	1 + $bs = 50K$	72.3M	27.84 (27.2)	⋆/1.00×	⋆/1.00×
9	1 + >12 layers + $bs = 25K \sim 50K$	-	-	-	-
10	4 + >12 layers + $bs = 25K \sim 50K$	-	-	-	
11	3 + 12 layers + $bs = 40K, dp_r = 0.3, dp_a = 0.2$	116.4M	28.27 (27.6)	102.9/1.00×†	0.188/1.00×†
12	11 + T2T	116.5M	28.03 (27.4)	107.7/0.96×†	0.191/0.98×†
13	11 + MAtt	103.8M	28.55 (27.9)	67.12/1.53×†	0.164 /1.15×†
14	3 + 20 layers + $bs = 44K, dp_r = 0.3, dp_a = 0.2$	175.3M	28.42 (27.7)	157.8/1.00×‡	0.283/1.00×‡
15	14 + T2T	175.3M	28.27 (27.6)	161.2/0.98×‡	0.289/0.98×‡
16	14 + MAtt	154.3M	**28.67** (**28.0**)	108.6/1.45×‡	0.251/1.13×‡

Table 3: Tokenized case-sensitive BLEU (in parentheses: sacreBLEU) on WMT14 En-De translation task. *#Param*: number of model parameters. △*Dec*: decoding time (seconds)/speedup on newstest2014 dataset with a batch size of 32. △*Train*: training time (seconds)/speedup per training step evaluated on 0.5K steps with a batch size of 1K target tokens. Time is averaged over 3 runs using Tensorflow on a single TITAN X (Pascal). "-": optimization failed and no result. "⋆": the same as model ①. † and ‡: comparison against ⑪ and ⑭ respectively rather than ①. *Base*: the baseline Transformer with base setting. Bold indicates best BLEU score. dp_a and dp_r: dropout rate on attention weights and residual connection. bs: batch size in tokens.

7.3 WMT14 En-De Translation Task

Table 3 summarizes translation results under different settings. Applying DS-Init and/or MAtt to Transformer with 6 layers slightly decreases translation quality by ∼0.2 BLEU (27.59→27.35). However, they allow scaling up to deeper architectures, achieving a BLEU score of 28.55 (12 layers) and 28.67 (20 layers), outperforming all baselines. These improvements can not be obtained via enlarging the training batch size (⑧), confirming the strength of deep models.

We also compare our simplified AAN in MAtt (④) with two variants: a self-attention network (⑥), and the original AAN (⑦). Results show minor differences in translation quality, but improvements in training and decoding speed, and a reduction in the number of model parameters. Compared to the baseline, MAtt improves decoding speed by 50%, and training speed by 10%, while having 9% fewer parameters.

Result ⑨ indicates that the gradient vanishing issue prevents training of deep vanilla Transformers, which cannot be solved by only simplifying the decoder via MAtt (⑩). By contrast, both T2T and DS-Init can help. Our DS-Init improves norm preservation through specific parameter initialization, while T2T reschedules the LN position. Results in Table 3 show that T2T underperforms DS-Init by 0.2 BLEU on average, and slightly increases training and decoding time (by 2%) compared to the original Transformer due to additional LN layers. This suggests that our solution is more effective and efficient.

ID	BLEU		PPL	
	Train	Dev	Train	Dev
1	28.64	26.16	5.23	4.76
11	29.63	26.44	**4.48**	**4.38**
12	**29.75**	26.16	4.60	4.49
13	29.43	**26.51**	5.09	4.71
14	30.71	26.52	**3.96**	**4.32**
15	**30.89**	26.53	4.09	4.41
16	30.25	**26.56**	4.62	4.58

Table 4: Tokenized case-sensitive BLEU (BLEU) and perplexity (PPL) on training (Train) and development (newstest2013, Dev) set. We randomly select 3K sentence pairs as our training data for evaluation. Lower PPL is better.

Surprisingly, training deep Transformers with both DS-Init and MAtt improves not only running efficiency but also translation quality (by 0.2 BLEU), compared with DS-Init alone. To get an improved understanding, we analyze model performance on both training and development set. Results in Table 4 show that models with DS-Init yield the best perplexity on both training and development set, and those with T2T achieve the best BLEU on the training set. However, DS-Init+MAtt performs best in terms of BLEU on the development set. This indicates that the success of DS-Init+MAtt comes from its better generalization rather than better fitting training data.

We also attempt to apply DS-Init on the encoder alone or the decoder alone for 12-layer models. Unfortunately, both variants lead to unstable optimization where gradients tend to explode at the

Task	Model	#Param	BLEU	△Dec	△Train
WMT14 En-Fr	Base + 6 layers	76M	39.09	167.56/1.00×	0.171/1.00×
	Ours + Base + 12 layers	108M	**40.58**	173.62/0.97×	0.265/0.65×
IWSLT14 De-En	Base + 6 layers	61M	34.41	315.59/1.00×	0.153/1.00×
	Ours + Base + 12 layers	92M	**35.63**	329.95/0.96×	0.247/0.62×
WMT18 En-Fi	Base + 6 layers	65M	15.5 (50.82)	156.32/1.00×	0.165/1.00×
	Ours + Base + 12 layers	96M	**15.8** (**51.47**)	161.74/0.97×	0.259/0.64×
WMT18 Zh-En	Base + 6 layers	77M	21.1	217.40/1.00×	0.173/1.00×
	Ours + Base + 12 layers	108M	**22.3**	228.57/0.95×	0.267/0.65×

Table 5: Translation results on different tasks. Settings for BLEU score is given in Section 7.1. Numbers in bracket denote chrF score. Our model outperforms the vanilla base Transformer on all tasks. "*Ours*": DS-Init+MAtt.

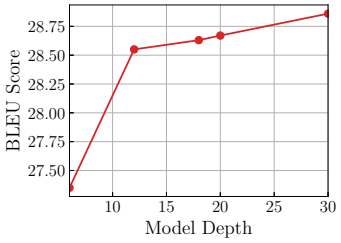


Figure 3: Test BLEU score on newstest2014 with respect to model depth for Transformer+DS-Init+MAtt.

Model	#Param	Test14	Test14-18
Vaswani et al. (2017)	213M	28.4	-
Chen et al. (2018)	379M	28.9	-
Ott et al. (2018)	210M	29.3	-
Bapna et al. (2018)	137M	28.04	-
Wu et al. (2019)	213M	**29.76**$_*$ (29.0)	33.13 $_*$ (32.86)
Big + 6 layers	233M	29.07 (28.3)	33.16 (32.88)
Ours + Big + 12 layers	359M	29.47 (28.7)	33.21 (32.90)
Ours + Big + 20 layers	560M	29.62 (29.0)	**33.26** (**32.96**)

Table 6: Tokenized case-sensitive BLEU (sacreBLEU) on WMT14 En-De translation task. "*Test14-18*": BLEU score averaged over newstest2014∼newstest2018. *: results obtained by running code and model released by Wu et al. (2019). "*Ours*": DS-Init+MAtt.

middle of training. We attempt to solve this issue with gradient clipping of rate 1.0. Results show that this fails for decoder and achieves only 27.89 BLEU for encoder, losing 0.66 BLEU compared with the full variant (28.55). We leave further analysis to future work and recommend using DS-Init on both the encoder and the decoder.

Effect of Model Depth We empirically compare a wider range of model depths for Transformer+DS-Init+MAtt with up to 30 layers. Hyperparameters are the same as for 14 except that we use 42K and 48K batch size for 18 and 30 layers respectively. Figure 3 shows that deeper Transformers yield better performance. However, improvements are steepest going from 6 to 12 layers, and further improvements are small.

7.3.1 Comparison with Existing Work

Table 6 lists the results in big setting and compares with current SOTA. Big models are trained with $dp_a = 0.1$ and $dp_r = 0.3$. The 6-layer baseline and the deeper ones are trained with batch size of 48K and 54K respectively. Deep Transformer with our method outperforms its 6-layer counterpart by over 0.4 points on newstest2014 and around 0.1 point on newstest2014∼newstest2018. Our model outperforms the transparent model (Bapna et al., 2018) (+1.58 BLEU), an approach for the deep encoder. Our model performs on par with current SOTA, the dynamic convolution model (DCNN) (Wu et al., 2019). In particular, though DCNN achieves encouraging performance on newstest2014, it falls behind the baseline on other test sets. By contrast, our model obtains more consistent performance improvements.

In work concurrent to ours, Wang et al. (2019) discuss how the placement of layer normalization affects deep Transformers, and compare the original *post-norm* (which we consider our baseline) and a *pre-norm* layout (which we call T2T). Their results also show that pre-norm allows training of deeper Transformers. Our results show that deep post-norm Transformers are also trainable with appropriate initialization, and tend to give slightly better results.

7.4 Results on Other Translation Tasks

We use 12 layers for our model in these tasks. We enlarge the dropout rate to $dp_a = 0.3, dp_r = 0.5$ for IWSLT14 De-En task and train models on WMT14 En-Fr and WMT18 Zh-En with 500K steps. Other models are trained with the same settings as in WMT14 En-De.

We report translation results on other tasks in Table 5. Results show that our model beats the baseline on all tasks with gains of over 1 BLEU,

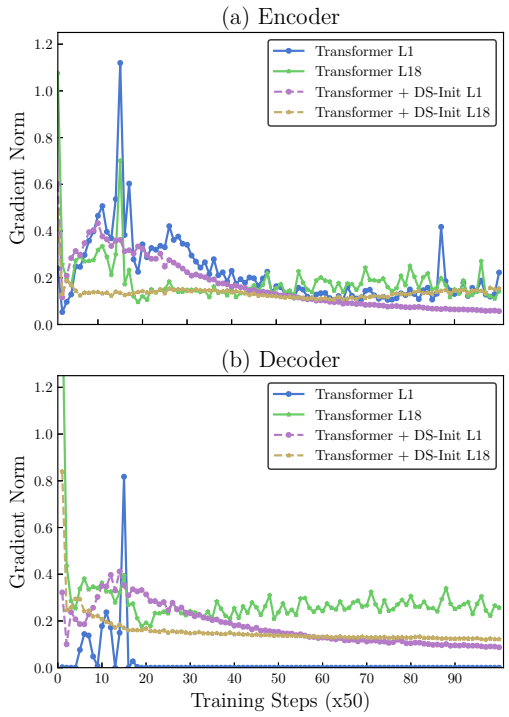


Figure 4: Gradient norm (y-axis) of the first and the last encoder layers (top) and decoder layers (bottom) in 18-layer deep Transformer over the fist 5k training steps. We use around 25k source/target tokens in each training batch. Each point in this plot is averaged over 50 training steps. "*L1*/*L18*" denotes the first/last layer. DS-Init helps stabilize the gradient norm during training.

except the WMT18 En-Fi where our model yields marginal BLEU improvements (+0.3 BLEU). We argue that this is due to the rich morphology of Finnish, and BLEU's inability to measure improvements below the word level. We also provide the chrF score in which our model gains 0.6 points. In addition, speed measures show that though our model consumes 50+% more training time, there is only a small difference with respect to decoding time thanks to MAtt.

7.5 Analysis of Training Dynamics

Our analysis in Figure 1 and Table 1 is based on gradients estimated exactly after parameter initialization without considering training dynamics. Optimizers with adaptive step rules, such as Adam, could have an adverse effect that enables gradient scale correction through the accumulated first and second moments. However, results in Figure 4 show that without DS-Init, the encoder gradients are less stable and the decoder gradients still suffer from the vanishing issue, particularly at the first layer. DS-Init makes the training more stable and robust.[6]

8 Conclusion and Future Work

This paper discusses training of very deep Transformers. We show that the training of deep Transformers suffers from gradient vanishing, which we mitigate with depth-scaled initialization. To improve training and decoding efficiency, we propose a merged attention sublayer that integrates a simplified average-based self-attention sublayer into the encoder-decoder attention sublayer. Experimental results show that deep models trained with these techniques clearly outperform a vanilla Transformer with 6 layers in terms of BLEU, and outperforms other solutions to train deep Transformers (Bapna et al., 2018; Vaswani et al., 2018). Thanks to the more efficient merged attention sublayer, we achieve these quality improvements while matching the decoding speed of the baseline model.

In the future, we would like to extend our model to other sequence-to-sequence tasks, such as summarization and dialogue generation, as well as adapt the idea to other generative architectures (Zhang et al., 2016, 2018). We have trained models with up to 30 layers each for the encoder and decoder, and while training was successful and improved over shallower counterparts, gains are relatively small beyond 12 layers. An open question is whether there are other structural issues that limit the benefits of increasing the depth of the Transformer architecture, or whether the benefit of very deep models is greater for other tasks and dataset.

Acknowledgments

We thank the reviewers for their insightful comments. This project has received funding from the grant H2020-ICT-2018-2-825460 (ELITR) by the European Union. Biao Zhang also acknowledges the support of the Baidu Scholarship. This work has been performed using resources provided by the Cambridge Tier-2 system operated by the University of Cambridge Research Computing Service (http://www.hpc.cam.ac.uk) funded by EPSRC Tier-2 capital grant EP/P020259/1.

[6] We observe this both in the raw gradients and after taking the Adam step rules into account.

References

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Ankur Bapna, Mia Chen, Orhan Firat, Yuan Cao, and Yonghui Wu. 2018. Training deeper neural machine translation models with transparent attention. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3028–3033, Brussels, Belgium. Association for Computational Linguistics.

Ondrej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. 2014. Findings of the 2014 workshop on statistical machine translation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA. Association for Computational Linguistics.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. 2018. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels. Association for Computational Linguistics.

Denny Britz, Anna Goldie, Minh-Thang Luong, and Quoc Le. 2017. Massive exploration of neural machine translation architectures. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1442–1451, Copenhagen, Denmark. Association for Computational Linguistics.

Mauro Cettolo, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT Evaluation Campaign, IWSLT 2014. In *Proceedings of the 11th Workshop on Spoken Language Translation*, pages 2–16, Lake Tahoe, CA, USA.

Mia Xu Chen, Orhan Firat, Ankur Bapna, Melvin Johnson, Wolfgang Macherey, George Foster, Llion Jones, Mike Schuster, Noam Shazeer, Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Lukasz Kaiser, Zhifeng Chen, Yonghui Wu, and Macduff Hughes. 2018. The best of both worlds: Combining recent advances in neural machine translation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 76–86. Association for Computational Linguistics.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers. *CoRR*, abs/1904.10509.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Mattia Antonino Di Gangi and Marcello Federico. 2018. Deep neural machine translation with weakly-recurrent units. In *Proceedings of EAMT*, Alicante, Spain.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N. Dauphin. 2017. Convolutional sequence to sequence learning. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1243–1252, International Convention Centre, Sydney, Australia. PMLR.

Felix A. Gers and Jürgen Schmidhuber. 2001. Long Short-Term Memory Learns Context Free and Context Sensitive Languages. In *Proceedings of the ICANNGA 2001 Conference*, volume 1, pages 134–137.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke S. Zettlemoyer. 2019. Constant-time machine translation with conditional masked language models. *CoRR*, abs/1904.09324.

Xavier Glorot and Y Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research - Proceedings Track*, 9:249–256.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor O.K. Li, and Richard Socher. 2018. Non-autoregressive neural machine translation. In *International Conference on Learning Representations*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780.

Marcin Junczys-Dowmunt, Kenneth Heafield, Hieu Hoang, Roman Grundkiewicz, and Anthony Aue. 2018. Marian: Cost-effective high-quality neural machine translation in C++. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation*, Melbourne, Australia.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. *arXiv preprint arXiv:1806.00187*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ngoc-Quan Pham, Thai-Son Nguyen, Jan Niehues, Markus Müller, and Alexander H. Waibel. 2019. Very deep self-attention networks for end-to-end speech recognition. *CoRR*, abs/1904.13377.

Matt Post. 2018. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence Level Training with Recurrent Neural Networks. In *The International Conference on Learning Representations*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

K. Simonyan and A. Zisserman. 2015. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*.

David R So, Chen Liang, and Quoc V Le. 2019. The evolved transformer. *arXiv preprint arXiv:1901.11117*.

Mitchell Stern, Noam Shazeer, and Jakob Uszkoreit. 2018. Blockwise parallel decoding for deep autoregressive models. In *Advances in Neural Information Processing Systems*, pages 10086–10095.

Ashish Vaswani, Samy Bengio, Eugene Brevdo, Francois Chollet, Aidan Gomez, Stephan Gouws, Llion Jones, Łukasz Kaiser, Nal Kalchbrenner, Niki Parmar, Ryan Sepassi, Noam Shazeer, and Jakob Uszkoreit. 2018. Tensor2Tensor for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Papers)*, pages 193–199, Boston, MA. Association for Machine Translation in the Americas.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

Mingxuan Wang, Zhengdong Lu, Jie Zhou, and Qun Liu. 2017. Deep neural machine translation with linear associative unit. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 136–145, Vancouver, Canada. Association for Computational Linguistics.

Qiang Wang, Bei Li, Tong Xiao, Jingbo Zhu, Changliang Li, Derek F. Wong, and Lidia S. Chao. 2019. Learning deep transformer models for machine translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1810–1822, Florence, Italy. Association for Computational Linguistics.

Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations*.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, ukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144.

Alireza Zaeemzadeh, Nazanin Rahnavard, and Mubarak Shah. 2018. Norm-preservation: Why residual networks can become extremely deep? *CoRR*, abs/1805.07477.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2018. Accelerating neural transformer via an average attention network. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1789–1798, Melbourne, Australia. Association for Computational Linguistics.

Biao Zhang, Deyi Xiong, and Jinsong Su. 2018. Neural machine translation with deep attention. *IEEE*

Transactions on Pattern Analysis and Machine Intelligence, pages 1–1.

Biao Zhang, Deyi Xiong, Jinsong Su, Hong Duan, and Min Zhang. 2016. Variational neural machine translation. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 521–530, Austin, Texas. Association for Computational Linguistics.

Hongyi Zhang, Yann N. Dauphin, and Tengyu Ma. 2019. Fixup initialization: Residual learning without normalization via better initialization. In *International Conference on Learning Representations*.

Xiangwen Zhang, Jinsong Su, Yue Qin, Yang Liu, Rongrong Ji, and Hongji Wang. 2018. Asynchronous bidirectional decoding for neural machine translation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Jie Zhou, Ying Cao, Xuguang Wang, Peng Li, and Wei Xu. 2016. Deep recurrent models with fast-forward connections for neural machine translation. *Transactions of the Association for Computational Linguistics*, 4:371–383.